Proceedings of AUVSI 2002, Orlando, FL, July 8 - 12, 2002.

Autonomous Mobility for the Demo III Experimental Unmanned Vehicles

Alberto Lacaze, Karl Murphy, Mark DelGiorno

Abstract— **This paper will describe the autonomous mobility controller used under the U.S. Army sponsored Demo III eXperimental Unmanned Vehicle (XUV) project. The controller is designed for avoiding obstacles and traversing highly unstructured outdoor environments. Behavior generation, world modeling and perception capabilities as well as design architecture will be presented. The architecture used is based on 4D-RCS which uses hierarchical levels of knowledge and decision making. At each successive level of the architecture, planning algorithms generate paths with increasing detail. High level commands developed by the soldiers are automatically refined using 30 meter DTED a priori maps and onboard sensors. The XUVs plan and replan their own routes, speed and gaze. Demo III XUVs have been thoroughly tested on scouting scenarios by the U.S. Army soldiers. The results of this testing will be presented.**

Keywords— **Planning, emerging behaviors, complexity, multiresolutional hierarchical control, Real-time Control Systems (RCS).**

I. Introduction

RELIABLE autonomous mobility in outdoor cross country environment presents a set of challenges that are not common to indoor systems. Specifically:
1. The support surface that the vehicle must traverse is not given and cannot assume to be flat as in indoor environments. Vegetation and water make it challenging to estimate support surface elevation.
2. Unlike indoor scenarios, a-priori knowledge of the terrain is not available or is poor in most cases.
3. Problems of localization and therefore registration are exacerbated by the fact that differential GPS is not available for many military applications.
4. Real-time sensing and scene interpretation are costly and produce large onboard computational and bandwidth requirements.
5. Classification of vegetation between traversable and non traversable is at large an unsolved problem.
6. Medium range sensing (50 to 200 meters) is very poor.

Deploying platforms in outdoor environments has the added challenge of the large amount of support personal needed to test, deploy, and maintain the robotic platforms. Demo III stands as one of the few robotics program that has not only funded the development of software and hardware needed for the development of outdoor robotic systems, but also, since its inception, emphasized the need to work in tough field conditions that mimic realistic military scenarios. This paper will present some of the results of this research program.

A. Lacaze and K. Murphy are with the Intelligent Systems Division of the National Institute for Standards and Technology. M. DelGiorno is with General Dynamics Robotics Systems.

II. Architecture

Planning algorithms have been in the literature for many years [1]. Hierarchical approaches to planning and representation have been studied under the label of operations research for even a longer period of time. There are undeniable advantages to organizing the control systems in hierarchical structures [2]. Some of these advantages include:
1. Reduction of computational burden.
2. Homogeneous resolution within each level and therefore easier representation.
3. Ability to re-plan at different rates at each level (slower replanning at the top of the hierarchy and faster replanning at the bottom of the hierarchy).
4. Distribution of computational burden between levels (i.e. distributed computing).
5. Efficient compression of representation.

In our case, we follow the RCS hierarchical architecture [3], [4]. RCS is a hierarchical control system divided into a multiplicity of levels of resolutions. At the top of the hierarchy, the representation is coarse with large time and space horizons and slow replanning cycles. At the bottom of the hierarchy the opposite is true. Fast replanning cycles at the bottom provide stability and quick response however the scope is small in time and space. In comparison with behavioral approaches, hierarchical systems tend to create a more explicit world representation. Cost criteria are used to evaluate a model of the system traversing the predicted world representation to achieve the assigned goal. First, a very coarse behavior is generated at the top of the hierarchy, and then this same behavior is refined at each level of the hierarchy. Proponents of hierarchical architectures argue that applying cost evaluation criteria is much easier to resolve using a complete representation as opposed to dealing with multi-

ple, sometimes contradicting, sets of behaviors. However, complex world representation and the complexity of testing plan combinations make the implementation of hierarchical systems challenging. RCS contains both reactive and deliberative (planning) components. Hierarchical architectures tend to lean toward planning solutions because they have a representation that allows the prediction components necessary for planning. Behavioral approaches tend to be more reactive in nature, which is sufficient in simple environments.

Within a hierarchy, sensing and perception loops tend to cluster the sensed data and, if needed, pass it to supervisor levels creating an upward flow of representation. On the other hand, commands tend to flow downward from coarse levels with large scopes, to the higher resolution levels with shorter scopes and faster control loops.

Figure 1 shows the outline of an RCS chain of command. Each level of the hierarchy (3 in the case of the Figure) is composed of three different functional entities: Sensory Processing (SP), World Model (WM) and Behavior Generation (BG).

Sensory Processing collects clusters and classifies data supplied by the sensors. The results of SP are sent to the WM for archival and registration. Low level SPs (bottom of the hierarchy) tend to deal with pixels and higher levels deal with discrete entities such as concepts that reflect several layers of sensor fusion.

World Model serves as a repository of SP results as well as servers to the BG modules. It collects registers and fuses information across time, as well as being the keeper of all the a-priori knowledge. Since it keeps the models of the level's working space, it predicts the outcome of behaviors generated by the GB module.

Behavior Generation hypothesizes sets of alternative plans that may satisfy the command sent from the level above, and sends them to the WM for evaluation. Plans that satisfy the requirements of the supervisor as well as cost criteria sent with the command are then sent to the surrogate levels for further refinement and execution.

There are many subtleties of RCS that help to reduce complexity. The modular approach encourages reusability of code at different levels [5], [6].

III. Autonomous Mobility

Figure 2 shows the different RCS levels used for autonomous vehicle control, as well as the different interfaces and suggestions for control cycle times. In this paper, we will concentrate on the Subsystem and Primitive levels for locomotion.

We assume that the Vehicle level will generate coarse

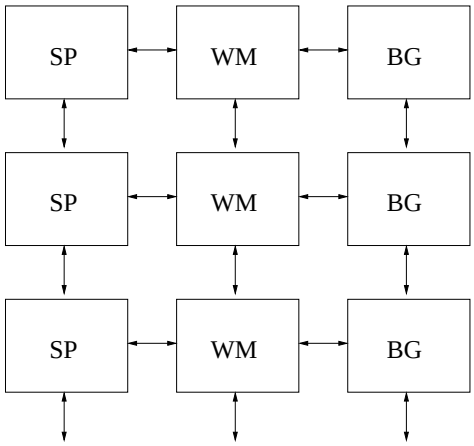

Fig. 1. Outline of an RCS hierarchy

Fig. 3. XUV mobility and sensor package

plans for the vehicle to follow, and that these paths will be sent to the Subsystem Level for execution. In general, these paths will have waypoints at about 50 meters, and be about 1000 meters in length. We also assume that the Servo Level will be able to follow the plans generated by the Primitive Level with a certain level of accuracy. The plans created by the Primitive Level will be about 20 meters in length and have 0.5 meter waypoints.

At the bottom of the RCS hierarchy are the sensors and actuators. The primary sensor on the XUV robot vehicle is a scanning laser range finder. It is mounted on a tilt head on the front right of the XUV. See Figure 3. This sensor provides about one hundred thousand measurements per second. To a range of about 60 meters. Other sensors include stereo vision (CCD and FLIR pairs) for medium range detection, navigation unit, GPS (Plugger), instrumented bumper, and instrumented suspension.

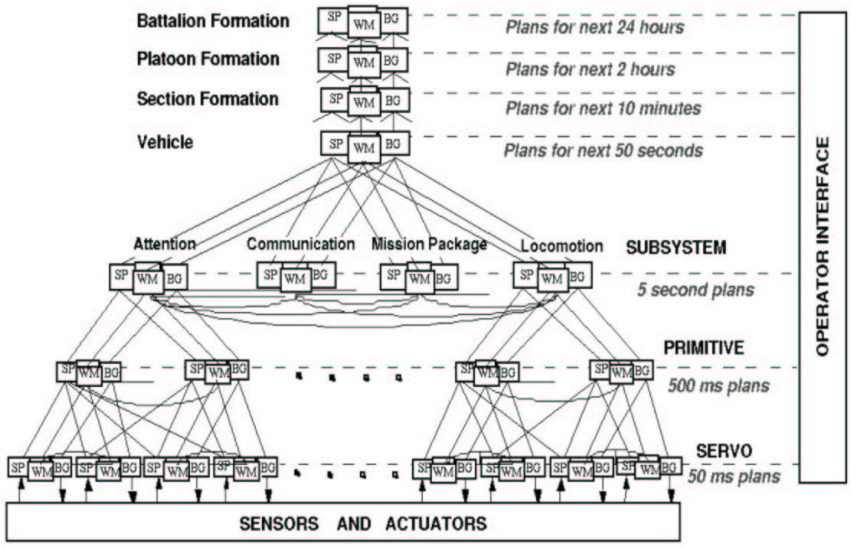

Fig. 2. RCS hierarchy for mobile platforms

IV. Support Surface

In order plan how to traverse the terrain in front of the vehicle, the BG modules need to be aware of the support surface. Determining the support surface is often a trivial task for indoor environments, however, in outdoor environments the presence of vegetation, dust, and water make the problem more difficult.

Our approach has the following steps:

1. LADAR range measurements are geometrically transformed into a voxel representation centered around the vehicle and collected as the vehicle move through the terrain.
2. The voxels record both "hits" and "transparencies". For each range measurement one hit occurs on laser beam at the measured distance. Where as several transparencies occur along every laser beam, starting at the LADAR and extending just short of the measured distance. Voxels with recorded transparencies are partially if not completely transparent.
3. A robust minimum (height) is used between the hits and transparencies counts to determine the support surface.
4. A robust ratio between the transparencies and hits is used to determine the "density" of a particular voxel. We assume that a less transparent voxel will be more likely to be solid than a transparent one. This is an assumption that it is not always true (i.e. glass, or steel wire). Therefore, other attributes from the color cameras, are used to better classify the traversability of the voxels.

Figure 4 shows the support surface found using the above described method for a terrain with shoulder height grass. The inverted cones show the path taken by the vehicle. The bottom of the cone is the location of the center of the navigation unit where wheels should meet the support surface. The cones are 1 meter high. The blue surface is the predicted support surface. The walls that cross the path of the vehicle are cross cuts of the voxel representation. This was done because if all voxels were displayed the support surface would not be viewable. The color of the cross cuts represents the number of hits (middle) and transparencies (bottom). The darker the red (green) color of the voxel, the more hits (transparencies) that particular voxel received. For most of the data collected, the error between the predicted support surface and the actual support surface does not exceed 0.25 meters for this kind of vegetation. The more dense the vegetation the harder it is for laser beams to penetrate, and therefore larger errors could be generated. The errors are monitored online as the vehicle compares predicted and actual elevation and makes adjustments to attitude and speed accordingly. Each voxel in the figure is 0.4 x 0.4 x 0.1 meters.

The voxel representation is only used at the Primitive Level since the number of hits available in the Subsystem Level is not large enough to determine an accurate support surface. The LADAR does not give consistent measurements of the horizontal plane past

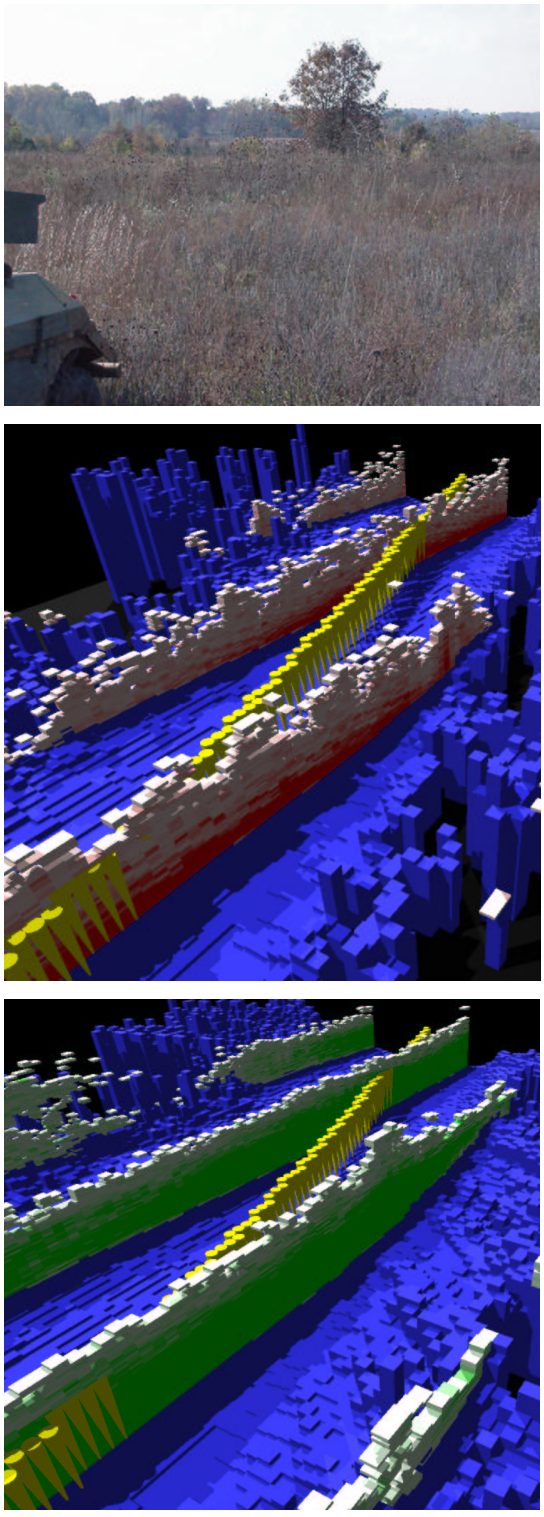

Fig. 4. Grassy field (top) voxel hits through field (middle) and transparencies (bottom) collected for an XUV traversing shoulder height vegetation

20 meters because of the shallow angle of incidence.

V. Planning Algorithm

A support surface is only calculated in the Primitive Level. Therefore, the strategies used for calculating the cost of traversal at the Primitive and Subsystem Levels are significantly different.

At the Primitive Level the support surface is used to determine the stability as well as roughness of the ride through several potential plans. At the Subsystem Level (past 20 meters) only sensed obstacles and a-priori data are used. The LADAR reasonably detects vertical obstacles up to 60 meters.

In both cases, the BG modules are graph based searches. [7] shows in more detail how these graphs are generated. Figure 5 shows the Primitive Level (curvy paths in the center) and the Subsystem Level (straight path in the outskirts) graphs. The top figure shows the graph used for an initial condition where the vehicle has the wheels pointed straight ahead, while the bottom picture shows the graph used when the vehicle has the wheels placed on the smallest turning radius.

A. Primitive Level

The Primitive Level graph computed offline (Figure 5) contains some of the parameters needed to calculate the cost to traverse each potential path. For example, the length and the side accelerations. Cost must also be added to account for traversing the local terrain. This includes the support surface and the "density" of the terrain that must be traversed to follow each trajectory. Therefore, the BG module uses the WM representation to evaluate the cost of traversal. Figure 6 shows how vehicle masks are placed along each of the Primitive Level trajectories. These masks are used to compute pitch and roll along each path. The masks are also used to determine compute amount of "density" that the vehicle must traverse to follow that trajectory. Other terms in the cost function include the roughness that each wheel will have to follow across the path, as well as checks for protruding objects that may hit the undercarriage.

Another important factor is whether a particular tile has already been seen. The cost evaluation will assign larger costs to trajectories that place the wheels of the vehicle in tiles that have never been seen by the sensor because these tiles have unknown elevation and may be holes or ditches. As these unknown elevation tiles get closer to the vehicle, the cost of the trajectories that cross them increases nonlinearly with distance.

All of these parameters are taken under consideration in order to calculate the costs of each trajectory.

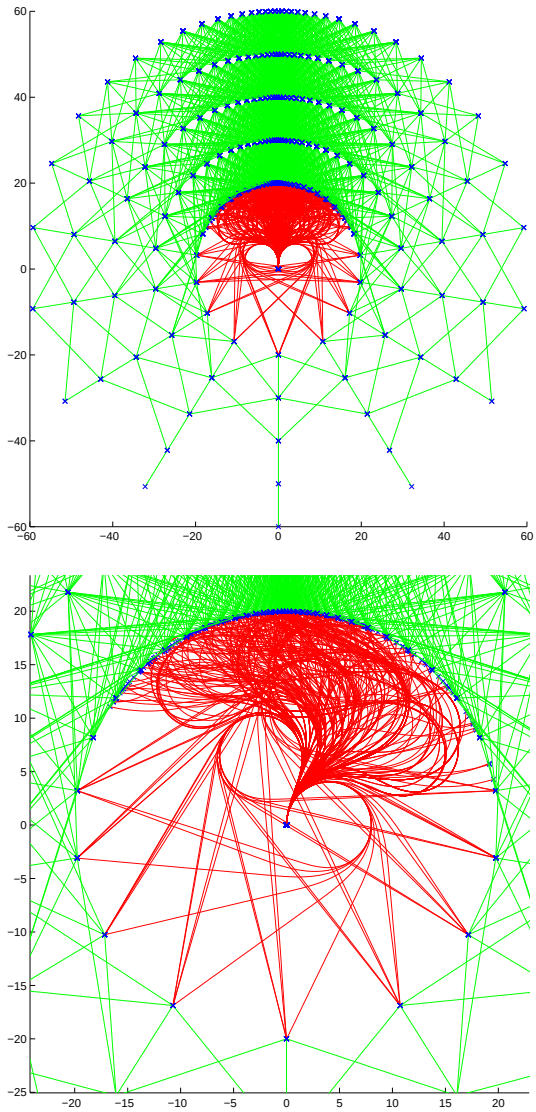

Fig. 5. Egographs for the Primitive and Subsystem Levels

They are computed as the search is underway. Therefore, trajectories (or edges of the graph) that are not opened by the search algorithm do not get evaluated. Replanning at this level is done at 4-10 Hz.

B. Subsystem Level

The Subsystem Level representation only contains obstacles and a priori data. In cases where only 30 meter DTED data is available, elevation is ignored as its accuracy is not sufficient for planning. The obstacles at this level are found by a cumulative probabilistic measure of the obstacles reported by the LADAR and stereo systems. The detections are accumulated in the

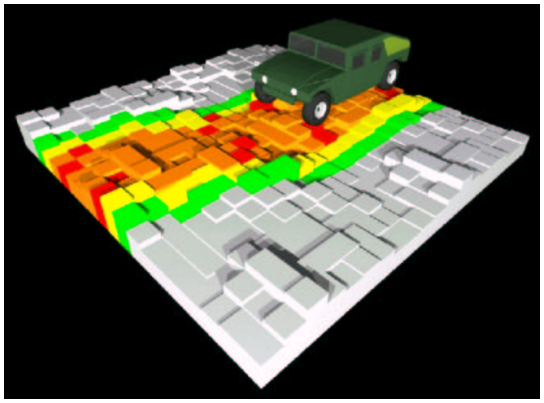

Fig. 6. Placing vehicle masks across the paths to compute cost

world model as the vehicle is moving through the terrain. The trajectories used by this level are straight line approximations. It is not necessary to generate kinematically correct trajectories since the Primitive level will have many opportunities (planning cycles) to refine the path selected by this level. The planner finds the optimal shortest obstacle free path available in the graph. The graph used by the Subsystem Level can be seen in Figure 5.

Figure 7 shows both levels at work. In this example, the XUV was crossing a bridge in a wooded area. The top figure shows the Vehicle Level plan (purple trajectory) that misses the bridge. This trajectory was done using 30 meters DTED. Because of the inaccuracies of the a priori data and the GPS drift, the Vehicle Level plan only marks the path coarsely across the bridge. The Subsystem and Primitive levels are given the freedom to deviate from the Vehicle Level plan to refine the path across the bridge. Blue areas on the Primitive Level on both pictures show areas that have not been seen by any of the sensors.

VI. Experimental Results

During Demo III, there were several test scenarios used to challenge the autonomy of the vehicles. Specifically, the vehicles:

1. The National Institute of Standards and Technology site in Gaithersburg Maryland was used to test and develop the planning algorithms on HMMWV platforms (inherited form the Demo II program) starting in the early 90's until today.
2. Aberdeen Proving Grounds was used to test the first XUVs systems. A demonstration run by U.S. Army soldiers was performed in relatively benign terrain (mowed grass) with discreet obstacles (rock walls, trees, and telephone posts). Safety, speed, as well as

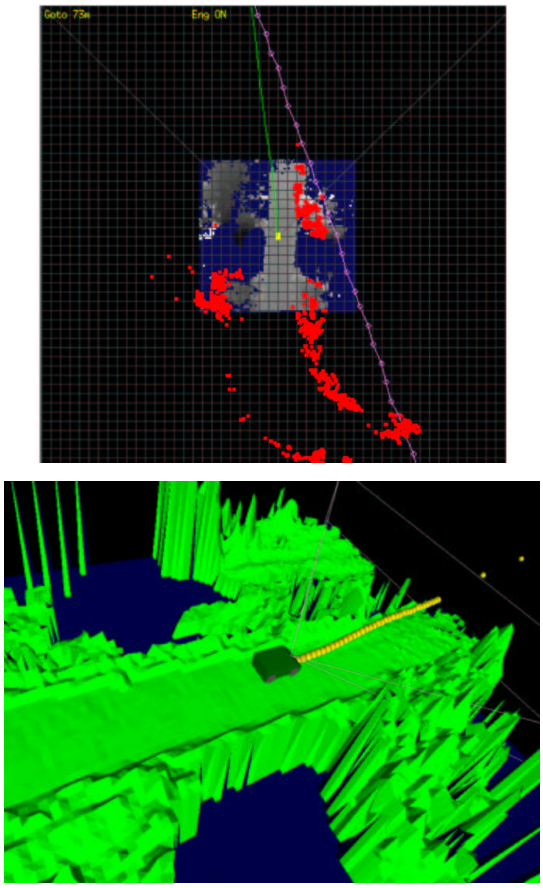

Fig. 7. Primitive and Subsystem Levels traversing a bridge. Top and 3d view.

reliability tests were performed at regular intervals.

3. The Fort Knox Tank Maneuver Area was used during Fall 2000 to test the vehicles in a larger area with significantly more challenging terrain. This area of Fort Knox is used to train tank drivers, and the terrain is full of ruts and is non-uniform. Two months were spent in field testing of the vehicles. The missions were run by U.S. Army soldiers. The vehicles were sent in one to two mile missions that included RSTA tasks. The terrain was significantly more challenging, and the vehicles showed competent behavior in open areas with difficult terrain features. The vehicles had problems traversing tall grass.

4. Fort Indiantown Gap Army Reserve Base has been used extensively for the past few years including Demo III in October 2001. This terrain includes a large variety of Appalachian foothill features. Hundreds of missions were accomplished by U.S. Army and U.S. Army Reserve soldiers. Most missions were about one mile in length and included areas of shoulder height grass. The behavior of the vehicles has significantly improved in heavy grass and vegetated areas by using the voxel representation. The performance of the vehicle at night positively compares with a human driver using NODs.

VII. Conclusions

- Outdoor mobility in highly unstructured environments has been demonstrated by the Demo III program.
- The challenges that can be found in traversing cross country are not the same that are available for indoor environments and therefore field tests are necessary to create realistic scenarios.
- RCS hierarchical architecture is well suited for outdoor mobility
- Novel support surface computations and planning algorithms have been developed to deal with the challenges of the environment.
- We believe that autonomous vehicles at this stage of development could be an asset for many applications.

References

[1] T. Perez-Lozano, "Spatial planning: A configuration space approach," *IEEE Trans. Computers*, vol. 32, 1983.

[2] Y. Maximov and A. Meystel, "Optimum design of multiresolutional hierarchical control systems," in *Proceedings of IEEE Int'l Symposium on Intelligent Control*, Glasgow, U.K., 1992, pp. 514–520.

[3] J.S. Albus, *Brain, Behavior, and Robotics*, McGraw-Hill, 1981.

[4] J. Albus, "An intelligent system architecture for manufacturing (ISAM)," Tech. Rep., NIST, 1996.

[5] J. Albus and A. Meystel, *Engineering of Mind*, Wiley series on Intelligent Systems, 2001.

[6] J. Albus and A. Meystel, *Intelligent Systems: Architecture, Design and Control*, Wiley Series on Intelligent Systems, 2002.

[7] A. Lacaze, Y. Moskovitz, N. DeClaris, and K. Murphy, "Path planning for autonomous vehicles driving over rough terrain," in *International Conference on Intelligent Systems*, NIST, 1998.